Sending User IDs to Sprig

In Sprig, you can utilize **User IDs** to link the participants in your surveys with the users in your own system. To ensure your team can effectively implement User IDs, here's what you need to know:

Helpful tips for using User IDs

- Ensure User IDs are mappable: The User IDs sent to Sprig should map back to your internal User IDs, either directly or anonymously.

- Ensure that User IDs are unique: User IDs should be unique within your population since Sprig leverages these User IDs to properly identify and survey your user population. Sprig allows User IDs to be formatted as any string or number.

- Ensure that User IDs do not change: User IDs should be static for a given user to ensure data consistency and to prevent surveying a user multiple times for a given survey.

- Anonymize User IDs: The User IDs you provide to Sprig are used exactly as you provide them. Although Sprig takes measures to encrypt and protect all user data, you are responsible for anonymizing any personally identifiable information before sending it to Sprig.

Unauthenticated & Authenticated Users

Sprig supports tracking unauthenticated users through the use of browser local storage. Most Sprig customers find this method effective when dealing with users that are not authenticated on their site. Keep in mind that this method is not completely reliable as users may be using multiple devices or blocking local storage.

Customers that allow users to authenticate with their site can provide Sprig with a User ID. When the Sprig system receives a User ID, it links up the

unauthenticated user to the now identified user, preserving any actions that might have been taken.

Anonymizing User IDs

We recommend you avoid identifying a user in Sprig with the exact User ID used in your internal database. Sprig takes steps to protect all user data via encryption, but you are still responsible for anonymizing any personally identifiable information before sending it to Sprig.

Here are 3 techniques you can use to anonymize your User IDs before sending them to Sprig:

1. Substitution using a lookup table

One option is to substitute the User ID (user_id) with another randomly generated UUID (anon_user_id). These UUIDs can be stored in a lookup table as reference keys to easily map back and forth between User IDs and anonymized User IDs. This can come in handy when linking Sprig data with your own internal user data.

user_id	anon_user_id
1	89967c78-72f0-4b35-b47d-ffd869172799
2	be76b585-4e20-4750-a042-4ce6532ab4b3
3	38f992a3-bc0e-4758-98ee-52a08b01c92e
...	...

2. Hashing

Hashing is a form of pseudonymization that can be used to transform a User ID into a fixed length obscure alphanumeric string, known as the hash value. This hash value can be used as an anonymous representation of your User ID. Hashing is intended to be a one-way data transformation process, so only use this technique if linking Sprig data with your internal user data is **not** a priority.

Here's a sample code snippet that can be used to transform a User ID into a pseudo-anonymized User ID in the node.js platform:

```
// Node.js user ID anonymization using an md5 hash
const crypto = require('crypto');
const anon_user_id =
   crypto.createHash('md5').update('user_id').digest('hex');
```

3. Symmetric key encryption

Symmetric key encryption uses a single secret key to convert personal data from 'plaintext' to 'ciphertext' which is unreadable without the knowledge of the secret key to reverse the encryption process. In this scenario, the User ID is encrypted in its 'ciphertext' representation, which can be used as the anonymous User ID.

When using encryption, proper care must be taken to ensure that the encryption protocol you select is secure, the secret key is generated randomly, and secret keys are stored securely. Please consult with your security team when determining the best encryption strategy for your use case.

Here's a sample code snippet for converting a User ID to its anonymized form using AES-256 symmetric key encryption:

```
// Node.js user ID encryption using AES-256 symmetric key encryption
const crypto = require('crypto');
const algorithm = 'aes-256-cbc';
const key = crypto.randomBytes(32);
const iv = crypto.randomBytes(16);

function encrypt(text) {
 const cipher = crypto.createCipheriv('aes-256-cbc', Buffer.from(key), iv);
 let encrypted = cipher.update(text);
 encrypted = Buffer.concat([encrypted, cipher.final()]);
 return { iv: iv.toString('hex'), encryptedData: encrypted.toString('hex') };
}
```

```
function decrypt(text) {
 const iv = Buffer.from(text.iv, 'hex');
 const encryptedText = Buffer.from(text.encryptedData, 'hex');
 const decipher = crypto.createDecipheriv('aes-256-cbc', Buffer.from(key),
iv);
 let decrypted = decipher.update(encryptedText);
 decrypted = Buffer.concat([decrypted, decipher.final()]);
 return decrypted.toString();
}

// Transform a user ID into its anonymized representation
const anon_user_id = encrypt('your user_id');

// Transform anonymized user ID back into the original user ID
const original_user_id = decrypt(anon_user_id);
```

Any questions about sending User IDs to Sprig?

The Sprig team is happy to help. Feel free to reach out to your dedicated Account Executive, or you can contact Sprig's Solutions Consultant Denisse Dominguez at denisse@sprig.com.